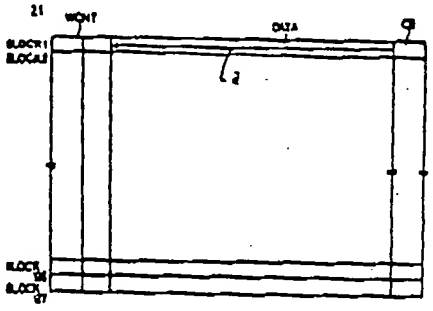
(54) MANAGEMENT SYSTEM FOR OF NUMBER OF TIMES OF WRITING PROGRAMMABLE READ ONLY MEMORY

(11) 62-283497 (A) (43) 9.12.1987 (19) JP
(21) Appl. No. 61-124732 (22) 31.5.1986
(71) CANON INC (72) SHINICHI NAKADA
(51) Int. Cl⁴. G11C17/00

PURPOSE: To average the rewriting frequency of a memory block and to prolong the life of an EEPROM by suppressing the writing to the memory block reaching the number of times of setting.

CONSTITUTION: The memory area of the EEPROM in which the erasing, the rewriting or the like are carried out by an input means, a CPU or the like is divided into 127 such as blocks BLOCK1~BLOCK127 and in the respective blocks, an area WCNT for storing the updating and rewriting number of times as well as a data memory area DATA is provided. When the contents of the area WCNT in which the counted value of the updating counter of the directory area of a block pointer is written are referred to and reach the set number, the rewriting of the block is suppressed through the CPU. The rewriting frequency of the respective blocks is averaged and the life of the EEPROM is prolonged.



a: storage data

⑲日本国特許庁(JP)　⑪特許出願公開

⑫公開特許公報(A)　昭62-283497

⑤Int.Cl.⁴	識別記号	庁内整理番号	㊸公開　昭和62年(1987)12月9日
G 11 C 17/00	307	6549-5B	

審査請求　未請求　発明の数 1　(全8頁)

㊹発明の名称　プログラマブルリードオンリメモリの書き込み回数管理方式

㉑特　願　昭61-124732

㉒出　願　昭61(1986)5月31日

㉒発明者　仲田　真一　東京都大田区下丸子3丁目30番2号　キヤノン株式会社内

㉑出願人　キヤノン株式会社　東京都大田区下丸子3丁目30番2号

㉔代理人　弁理士　小林　将高

明　細　書

1．発明の名称

プログラマブルリードオンリメモリの書き込み回数管理方式

2．特許請求の範囲

記憶領域に書き込まれた情報を電気的に消去可能なプログラマブルリードオンリメモリにおいて、前記記憶領域を複数のブロックに分割し、各ブロック毎に書き込み回数を記憶し、あらかじめ設定される書き込み回数を越えたブロックへの書き込みを抑止させることを特徴とするプログラマブルリードオンリメモリの書き込み回数管理方式。

3．発明の詳細な説明

〔産業上の利用分野〕

この発明は、電気的消去可能なプログラマブルリードオンリメモリの書き込み回数の管理方式に関するものである。

〔従来の技術〕

従来のＥＥＰＲＯＭ（Electrical Eresable and Programmable ROM）は、容量も少なく、また書き込むために必要な外部回路が多かった。さらに、チップ内のすべてのデータを消去するモードしか有していなかった。最近は、容量も大きくなるとともに、外部回路も殆ど必要なくＣＰＵのアドレスバス，データバスに結線できるようになり、またＥＥＰＲＯＭ内の１バイトのデータのみの消去も可能となってきた。以上の改良により、使用目的によっては、従来のランダムアクセスメモリ（ＲＡＭ）で構成していた機能の置換が可能となった。

例えば、従来の小型パソコン、日本語ワープロで作成したプログラムや文章，外字等を保存しておくためにメモリカードと云うものがある。これは、必要なときにパソコン，日本語ワープロ等の本体に差し込んでプログラムや文章を記憶させ、本体から引き抜いても、そのデータを記憶しているように、メモリカード内にはＲＡＭと電池が搭載されていた。そこで、メモリカードをＥＥＰＲＯＭで構成することにより、電池を無くすること

ができると考えられた。

〔発明が解決しようとする問題点〕

ところが、ＥＥＰＲＯＭでは従来のＲＡＭのように自由に何度も書き換えられない制約があり、すなわち、あらかじめ設定される書き込み回数を越えて、メモリカードへの書き込みを行なうことにより、記憶しているはずのデータを消失させてしまう等の問題点があった。またＥＥＰＲＯＭに書き込まれたデータのうち、頻繁に書き換えられるデータと書き換え頻度の少ないデータとが存在し、書き換え頻度の高いデータの書き換え回数が所定値を越えると、ＥＥＰＲＯＭへの書き換えが可能にも関わらず書き換え不能となる問題点があった。

この発明は、上記の問題点を解消するためになされたもので、ＥＥＰＲＯＭに書き込まれるデータの消失を防止するとともに、ＥＥＰＲＯＭへの書き込み回数を平均化させるとともに、ＥＥＰＲＯＭ上の書き換え頻度を平均化して、ＥＥＰＲＯＭへの書き換え寿命を延命できるプログラマブルリードオンリメモリの書き込み回数管理方式を得ることを目的とする。

〔問題点を解決するための手段〕

この発明に係るプログラマブルリードオンリメモリの書き込み回数管理方式は、記憶領域を複数のブロックに分割し、各ブロック毎に書き込み回数を記憶し、あらかじめ設定される書き込み回数を越えたブロックへの書き込みを抑止させる。

〔作用〕

この発明においては、記憶領域を各ブロック毎に書き込み回数を記憶しておき、この書き込み回数があらかじめ設定される書き込み回数を越えたら、そのブロックへの書き込みを抑止させる。

〔実施例〕

第１図（ａ）はこの発明の一実施例を示すプログラマブルリードオンリメモリへの書き込み回数管理方式を説明する模式図であり、１はＥＥＰＲＯＭで、例えば書き込み容量が32798 バイト×8 ビットで、書き込み回数が１万回に設定してある。ＥＥＰＲＯＭ１は、ポインタブロック１ａおよび予備ポインタブロックＳＰＢ１～ＳＰＢ５０より構成される。ポインタブロック１ａは４アドレス（各１バイト）で構成され、『０～１』番地の２バイトで、書き換え回数ＷＣＮＴ、例えば『１３８８$_{16}$』を記憶している。またポインタブロック１ａの『２』番地の１バイトは、ディレクトリＤＢ、例えば『０１$_{16}$』を記憶している。さらに、ポインタブロック１ａの『３』番地の１バイトは、未使用のスタートブロック番号ＯＳＢ、例えば『３３$_{16}$』を記憶している。またポインタブロック１ａの『４』番地の１バイトは、未使用のエンドブロック番号ＯＥＢ、例えば『８Ａ$_{16}$』を記憶している。

第１図（ｂ）はこの発明の装置構成の一例を説明するブロック図であり、１１はＣＰＵで、ＲＯＭ１１ａ，ＲＡＭ１１ｂを有し、ＲＯＭ１１ａに格納された第６図に示すフローに沿ったプログラムに応じて各部を制御する。１２は入力手段で、データ書き込み装置１３にセットされるＥＥＰＲＯＭ１へのデータ書き込みおよびデータ消去を指示する。なお、ＣＰＵ１１にはデータの転送を行うアキュムレータＡＣＣ，ＢＣＣを有している。

第２図は第１図（ａ）に示すＥＥＰＲＯＭ１の構造を示す模式図であり、２１はブロック番号であり、例えば１２７個のブロックＢＬＯＣＫ１～ＢＬＯＣＫ１２７に分割されている。各ブロックは、例えば２５６バイトで構成され、先頭の２バイトで、そのブロックが更新された回数、すなわち、後述する更新回数が記憶されている。次に続く２５３バイトは記憶データＤＡＴＡが記憶されており、最後の１バイトは、記憶データＤＡＴＡがこのブロックに留まるか、または他のブロックに及ぶかどうかを示す継続ブロックエリアＣＢがあり、他のブロックに記憶データＤＡＴＡが及ぶ場合は、継続ブロックエリアＣＢには継続するブロック番号が記憶され、他のブロックに記憶データＤＡＴＡが及ばない場合は、継続ブロックエリアＣＢには『ＦＦ$_{16}$』が記憶されている。

第３図は第２図に示すディレクトリブロック構造を説明する模式図であり、３０は前記ディレ

クトリＤＢに指示されるディレクトリブロック，３１は前記ディレクトリブロック３０の更新カウンタで，例えば２バイトで構成される．３２はファイル領域で，各ファイル名が１２バイトで記憶される．３３はスタートブロック番号エリア（ＳＢ）で，例えば１バイトで構成され，ファイルのスタートブロック番号が記憶されている．３４はエンドブロック番号エリア（ＥＢ）で，例えば１バイトで構成され，ファイルのエンドブロック番号が記憶されている．３５はチェーンブロックエリア（ＣＢ）で，ディレクトリブロック３０に継続するディレクトリブロックの有無を記憶する．例えばチェーンブロックエリア３５が『ＦＦ$_{16}$』となる．なお，ディレクトリブロック３０は，例えば１８個のファイル領域３２で構成される．

次に第１図（ａ）および第３図を参照しながらＥＥＰＲＯＭ１の構造について説明する．

第１図（ａ）に示すようにポインタブロック１ａの書き換え回数ＷＣＮＴに，例えば『１３８８$_{16}$』が記憶されているとすると，５０００回の更新が行われたことを示し，またディレクトリＤＢには『０１$_{16}$』が記憶されているので，ディレクトリＤＢに指示されるディレクトリブロック３０のブロック番号が『１』で，そのディレクトリブロック３０の更新カウンタ３１には，『１４２Ｆ$_{16}$』が記憶されている．これは，このディレクトリブロック３０を５１６７回更新したことを示し，ファイル領域３２のファイル（ＦｉＬｅ）１（ファイル名）はスタートブロック番号エリア３３が『０２$_{16}$』で，エンドブロック番号エリア３４が『０５$_{16}$』となっているため，ブロックＢＬＯＣＫ２から始まり，ブロックＢＬＯＣＫ５で終ることになる．またファイル領域３２のファイル２は，スタートブロック番号エリア３３が『０Ａ$_{16}$』で，エンドブロック番号エリア３４が『０Ｆ$_{16}$』となっているため，ブロックＢＬＯＣＫ１０から始まり，ブロックＢＬＯＣＫ１５で終ることになる．さらに，ファイル領域３２のファイル３（ファイル名）は，スタートブロック番号エリア３３が『１５$_{16}$』で，エンドブロック番号エリア３４が『１８$_{16}$』となっているため，ブロックＢＬＯＣＫ２１から始まり，ブロックＢＬＯＣＫ２４で終ることになる．またファイル領域３２のファイル３の次に『ＦＦ$_{16}$』が書かれているので，このファイル領域３２はファイル３で終了していることになる．

第４図は未使用のＥＥＰＲＯＭ１の状態を説明する模式図であり，第１図（ａ），第３図と同一のものには同じ符号を付している．

この図から分かるように，未使用のＥＥＰＲＯＭ１のポインタブロック１ａの書き換え回数ＷＣＮＴが『０００１$_{16}$』，ディレクトリＤＢが『０１$_{16}$』，未使用のスタートブロック番号ＯＳＢが『０２$_{16}$』，未使用のエンドブロック番号ＯＥＢが『７Ａ$_{16}$』がそれぞれポインタブロック１ａの０番地から４番地にそれぞれ記憶されている．これにより，ディレクトリＤＢに指示されるブロックＢＬＯＣＫ１を参照すると，更新カウンタ３１に『０００１$_{16}$』が書き込まれているとともに，ファイル領域３２のファイル１に『ＦＦ$_{16}$』が書き込まれており，さらに，チェーンブロックエリア３５に『ＦＦ$_{16}$』が書き込まれており，ＥＥＰＲＯＭ１が未使用状態であることを示している．

さらに，ポインタブロック１ａのスタートブロック番号ＯＳＢおよびエンドブロック番号ＯＥＢには『０２$_{16}$』，『７Ｆ$_{16}$』がそれぞれ書き込まれている．すなわち，ブロックＢＬＯＣＫ２～１２７には先頭の２バイトに『０００１$_{16}$』が書き込まれ，最後の１バイトに各後続のブロックの継続を示すチェーンブロックエリア３５には，ブロックＢＬＯＣＫ２～１２８に対して『０３～７Ｆ$_{16}$』が書き込まれ，ブロックＢＬＯＣＫ１２７のチェーンブロックエリア３５には『ＦＦ』が書き込まれている．このように，各ブロックＢＬＯＣＫ２～１２７は１つのチェイン構造となる．

次に第３図，第５図（ａ），（ｂ）を参照しながらＥＥＰＲＯＭ１への書き込み動作を説明する．

第５図（ａ），（ｂ）はＥＥＰＲＯＭ１への書き込み動作を説明する模式図であり，第１図

（ａ），第３図と同一のものには同じ符号を付している。なお、書き込み直前は、第３図に示す状態であったものとする。

まず、各ブロックＢＬＯＣＫのファイル領域３２の先頭が『００$_{16}$』のところを探し当てる。第３図の場合は、ファイル２とファイル３との間に『００$_{16}$』があり、そこにファイル４という名前を１２バイトで書き込み、ポインタブロック１ａの未使用ブロックのスタートブロック番号ＯＳＢを参照して、スタートブロック番号ＯＳＢの指示するブロックＢＬＯＣＫ、すなわち『５７$_{16}$』の先頭の２バイト情報、すなわち、更新カウンタ３１を『１』インクリメントし、その加算値が、例えば１万回を越えているようであれば、ファイル４のチェーンブロックエリア３５が示すブロックＢＬＯＣＫに対して同様の操作を行い、更新カウンタ３１が１万回以下のブロックＢＬＯＣＫを探し当てて、そのブロックＢＬＯＣＫの番号をポインタブロック１ａのスタートブロック番号ＯＳＢに書き込むとともに、ファイル４のデータをブロックＢＬＯＣＫ８７（２５３バイト）に書き込み、ブロックＢＬＯＣＫ８７に越れるようであれば、ブロックＢＬＯＣＫ８７のチェーンブロックエリア３５の指示するブロックＢＬＯＣＫの更新カウンタ３１を『１』インクリメントして加算値が、例えば１万回を越えているかどうかを調べ、指示されるブロックＢＬＯＣＫの更新カウンタ３１が１万回を越えるようであれば、更新回数が１万回以下のブロックＢＬＯＣＫを探し当て、そのブロックＢＬＯＣＫの番号を直前に書き込んだブロックＢＬＯＣＫのチェーンブロックエリア３５に書き込む。このようにして、データの書き込みが行われ、更新回数が１万回を越えるブロックＢＬＯＣＫが排除されて行く。そして、書き込みデータがなくなるまで同様の操作を行い、最後に書き込んだブロックＢＬＯＣＫのチェーンブロックエリア３５に記憶されていた内容を新しい未使用のスタートブロック番号ＯＳＢに書き換え、ポインタブロック１ａの書き換え回数ＷＣＮＴを『１』インクリメントして『１３８９$_{16}$』となり、最後にデータを書き込んだブロックＢＬＯＣＫのチェーンブロックエリア３５を『ＦＦ$_{16}$』にする。そして、ディレクトリブロック３０の最終ブロック番号を記憶するエンドブロック番号エリア３４に最後のデータを書き込んだブロックＢＬＯＣＫの番号を書き込むとともに、更新カウンタ３１を『１』インクリメントすると、第５図（ｂ）に示されるように、更新カウンタ３１が『１４３０$_{16}$』となり、ファイル４のスタートブロック番号エリア３３が『３３$_{16}$』で、エンドブロック番号エリア３４が『３７$_{16}$』となる。

次に第５図（ａ），（ｂ）を参照しながらＥＥＰＲＯＭ１に書き込まれているファイル１の削除動作について説明する。

ディレクトリブロック３０となるブロックＢＬＯＣＫ１よりファイル１を探し、ファイル領域３２の先頭の２バイトを『００$_{16}$』とする。次いで、ディレクトリブロック３０の更新カウンタ３１を『１』インクリメントし、ファイル１のスタートブロック番号エリア３３とエンドブロック番号エリア３４のデータを参照して、ポインタブロック１ａのエンドブロックＯＥＢが指示するブロックのチェーンブロックエリア３５の内容（削除直前までは『ＦＦ$_{16}$』であった）をスタートブロック番号エリア３３の内容に変更し、このブロックの更新カウンタ３１を『１』インクリメントする。すなわち、未使用ブロックの最後に今削除したファイル４を接続するわけである。このようにして、更新カウンタ３１を進めながら何度もファイルの更新，削除を実行して行くうちに、更新カウンタ３１が１万回に接近する。

次に更新カウンタ３１が１万回に到達した場合のアクセス処理について説明する。

まず、ポインタブロック１ａのスタートブロック番号ＯＳＢの内容が示しているブロックＢＬＯＣＫのチェーンブロックエリア３５の内容を新規のスタートブロック番号ＯＳＢとする。次いで、このブロック直前のディレクトリブロック３０の更新カウンタ３１の情報以外の内容を転送する。そして、ポインタブロック１ａのディレクトリＤ

Bに新規のディレクトリブロック番号を書き込み、ポインタブロック1aの書き換え回数WCNTおよび更新カウンタ31を「1」インクリメントする。

一方、ポインタブロック1aの書き換え回数WCNTは1万回を越えた場合は、予備ポインタブロックSPB1～SPB50のうち一番近い予備ポインタブロックへ書き換え回数WCNTの情報以外のデータを転送し、新規のポインタブロックの書き換え回数WCNT（0000_{16}）を「1」インクリメントして「0001_{16}」に設定する。この場合、破棄されたポインタブロック1aの書き換え回数WCNTは1万回以上となり、新のポインタブロック1aの書き換え回数WCNTは1万回以下となる。このようにして、カウンタブロック30およびポインタブロック1aの書き込み削除を管理する。また削除されたファイルが使用していたブロックは未使用ブロックの一番最後に回される。これは、未使用ブロックの使用回数を平均化するためである。

第6図は第1図（a）に示したEEPROM1のデータ書き込み制御動作を説明するためのフローチャートである。なお、(1)～(18)はるステップを示す。

まず、ディレクトリブロック30の空エリアを探して、新規のファイル名を書き込む(1)。次いで、未使用のスタートブロック番号OSBをCPU11のアキュムレータACCに記憶させる(2)。アキュムレータACCが指示するブロックの書き換え回数WCNTを＋1更新する(3)。ここで、書き換え回数WCNTが10000を越えたかどうかを判断し(4)、YESならばアキュムレータACCの指示するブロックの継続ブロックエリアCBをアキュムレータACCに記憶し(5)、ステップ(3)に戻り、NOならばディレクトリブロック30のスタートブロック番号エリア（SB）33にアキュムレータACCの内容を書き込む(6)。次いで、アキュムレータACCが指示するブロックのデータエリアにデータを書き込む(7)。ここで、書き込みデータがアキュムレータACCが指示するブロックの容量が235バイトを越えるかどうかを判断し(8)、YESならばアキュムレータACCが指示するブロックの継続ブロックエリアCBをアキュムレータBCCに記憶させる(9)。次いで、アキュムレータBCCが指示するブロックの書き換え回数WCNTを＋1更新する(10)。次いで、書き換え回数WCNTが10000を越えたかどうかを判断し(11)、YESならばアキュムレータBCCの指示するブロックの継続ブロックエリアCBを記憶させ(12)、ステップ(10)に戻り、NOならばアキュムレータACCが指示するブロックの継続ブロックエリアCBにアキュムレータBCCの内容を書き込み(13)、ステップ(7)に戻る。

一方、ステップ(8)の判断でNOの場合は、アキュムレータACCが指示する継続ブロックエリアCBを未使用のスタートブロック番号OSBに書き込む(14)。次いで、ポインタブロック1aの書き換え回数WCNTを＋1更新する(15)。次いで、アキュムレータACCが指示するブロックの継続ブロックエリアCBへ「FF_{16}」を書き込む(16)。そして、ディレクトリブロック30の新ファイル位置のエンドブロック番号エリア34へアキュムレータACCの内容を書き込む(17)。次いで、ディレクトリブロック30の書き換え回数WCNTを更新する(18)。

（発明の効果）

以上説明したように、この発明は記憶領域を複数のブロックに分割し、各ブロック毎に書き込み回数を記憶し、あらかじめ設定される書き込み回数を越えたブロックへの書き込みを抑止させるようにしたので、EEPROMに書き込まれるデータの消失を未然に防げるとともに、不要になったブロックを未使用ブロックの最後尾に接続するようにしたので、各ブロックの書き込み回数を平均化できる利点を有する。

4．図面の簡単な説明

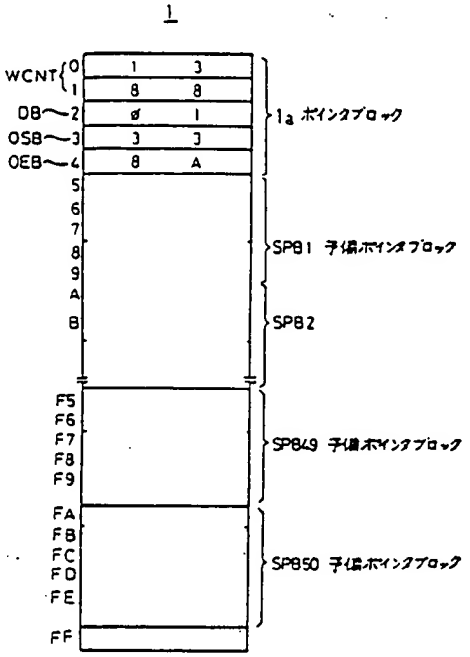
第1図（a）はこの発明の一実施例を示すプログラマブルリードオンリメモリへの書き込み回数管理方式を説明する模式図、第1図（b）はこの

発明の装置構成の一例を説明するブロック図、第２図は第１図（ａ）に示すＥＥＰＲＯＭの構造を示す模式図、第３図は第２図に示すディレクトリブロック構造を説明する模式図、第４図は未使用のＥＥＰＲＯＭ状態を説明する模式図、第５図（ａ），（ｂ）はＥＥＰＲＯＭへの書き込み動作を説明する模式図、第６図は第１図（ａ）に示したＥＥＰＲＯＭのデータ書き込み動作を説明するためのフローチャートである。

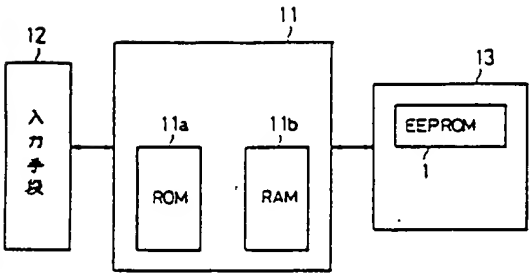
図中、１はＥＥＰＲＯＭ、１ａはポインタブロック、２１はブロック番号、３０はディレクトリブロック、３１は更新カウンタ、３２はファイル領域、３３はスタートブロック番号エリア、３４はエンドブロック番号エリア、３５はチェーンブロックエリアである。

代理人　小　林　将　高

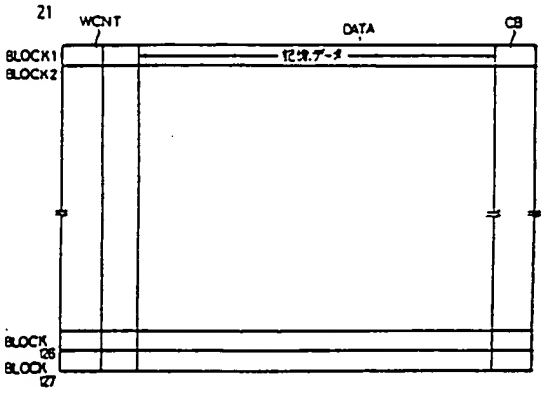
第　１　図　(a)



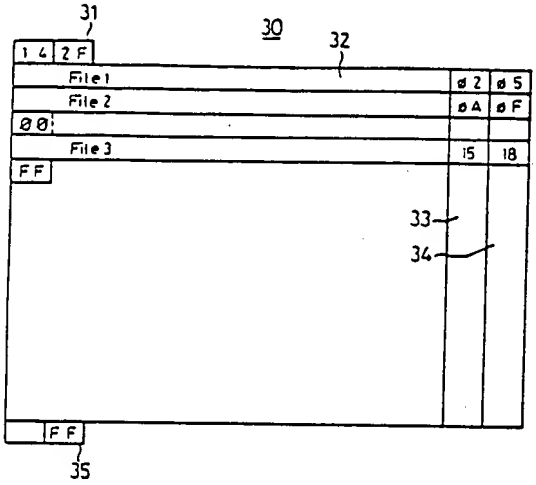
第　１　図　(b)



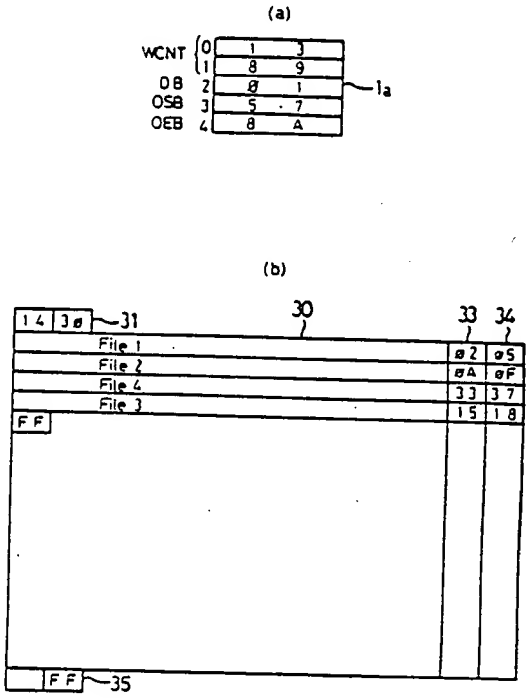
第　２　図



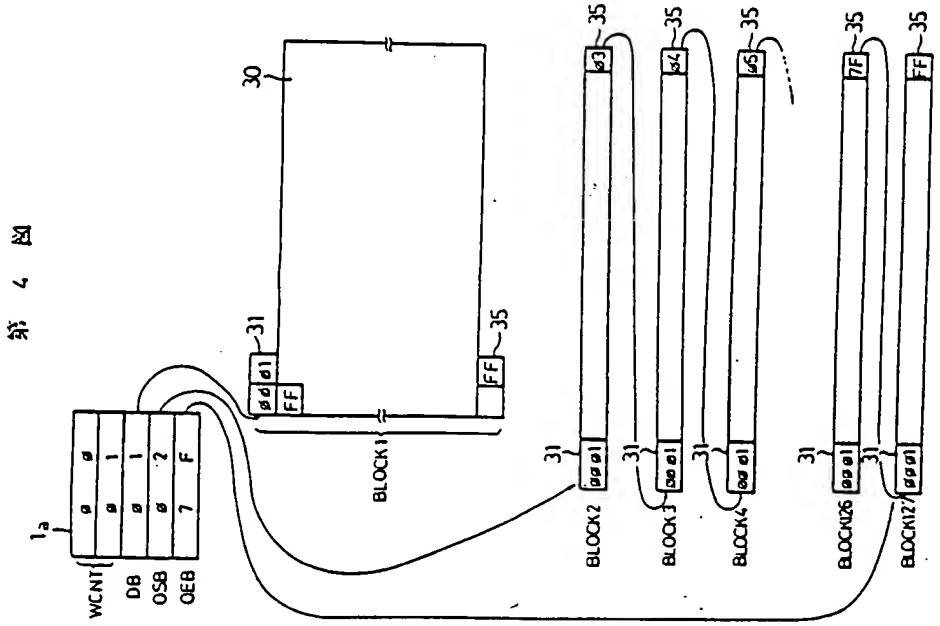
第 3 図



第 5 図



第 4 図



第 6 図